Eurographics Symposium on Rendering 2026
I. Gkioulekas and A. Jarabo
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 4

Neural Render Proxies for Interactive and Differentiable Lighting: Supplemental Material

Sergio Sancho[1,2], Alexander Rath[2], Marco Manzi[2], Pascal Chang[1,2]
Amit H. Bermano[1,2,3], Derek Nowrouzezahrai[4,5,6], Markus Gross[1,2], Marios Papas[2]

[1]ETH Zurich, Switzerland [2]DisneyResearch|Studios, Switzerland [3]Tel Aviv University, Israel
[4]McGill University, Canada [5]Mila – Quebec AI Institute [6]CIFAR AI Chair

Appendix A: Additional Experiments

Effect of Sample Count

Table 5 reports standard image quality metrics (relMSE, SMAPE, FLIP, PSNR, and SSIM) across different training configurations. Specifically, we ablate the samples per pixel (spp) used to reconstruct the training target images, noting that a higher spp requires recording more path data from the renderer. For each metric, we compare our trained MLP's predictions against reference images denoised at the corresponding spp.

We report the median metric values across 2,000 images, each generated with a unique lighting configuration (varying light position and radius). The ground truth images used for these evaluations are rendered in Mitsuba at 2048 spp and subsequently denoised. All evaluated models feature an 8-layer MLP with 256 neurons per layer and are trained for 100k steps.

We provide extended metrics (MSE, Tonemapped MSE, and LogMSE) in Tables 8 and 9 for our model.

Effect of Network Size

Similarly, we evaluate the impact of network capacity (number of neurons $\times$ number of layers) while holding the training data reconstruction fixed at 256 spp. Table 10 details the median quantitative metrics across all 2,000 lighting configurations (also shown in Figure 1).

The results demonstrate that while reconstruction quality consistently improves as model size increases, these performance gains eventually plateau for the largest networks.

Timings & Memory Usage

We evaluate the computational performance and memory footprint across the core components of our method. Table 1 reports the network inference times for varying capacities (layers and neurons), while Table 2 details the image reconstruction and OIDN [Áfr26] denoising times across different spp settings. The reconstruction utilizes our fused Triton kernel with fp16+rgb9e5 precision. In all tables, our default configurations are highlighted in bold. Additionally, Table 3 outlines the GPU memory required to load path data during the reconstruction phase.

Finally, we profile the training performance across varying spp levels and network sizes on an NVIDIA RTX 5090. Table 6 and Table 7 present the absolute time and relative percentage of each operation within a training iteration, alongside the estimated and empirically observed total time for 100k iterations. The final column in these tables reports the peak GPU memory consumption.

Our profiling reveals that backpropagation dominates the training iteration. Notably, generating new images to update the training pool ("Pool") accounts for only $\sim$19% of the iteration time under our default configuration (8 layers of 256 neurons, 512 spp), underscoring the efficiency of our optimized reconstruction kernel. Furthermore, compressing the path data with rgb9e5 ensures that even our most demanding configurations fit comfortably within the 32 GB memory limit of a consumer-grade GPU.

Table 1: Inference time (ms) by network size

Network	Bedroom	Cornell Box	Kitchen
16×4	0.82 ± 0.01	0.86 ± 0.00	0.83 ± 0.00
32×4	0.96 ± 0.00	1.01 ± 0.01	0.98 ± 0.06
64×4	1.20 ± 0.00	1.25 ± 0.01	1.20 ± 0.00
128×8	3.12 ± 0.03	3.28 ± 0.03	3.13 ± 0.03
256×8	**9.12 ± 0.08**	**9.61 ± 0.08**	**9.13 ± 0.08**
512×8	31.09 ± 0.05	32.85 ± 0.04	31.21 ± 0.03

Convergence Analysis

Figure 2 and Figure 3 illustrate the evolution of several quality metrics throughout training, evaluated across multiple intermediate checkpoints. We plot these metrics with respect to both training steps and estimated wall-clock time, utilizing logarithmic scales where appropriate to better highlight convergence trends. Notably,

Table 2: Reconstruction time (ms) by spp

spp	Bedroom	Cornell Box	Kitchen
32	1.04 ± 0.04	0.98 ± 0.01	1.03 ± 0.05
64	1.91 ± 0.01	1.70 ± 0.01	1.85 ± 0.01
128	3.59 ± 0.01	3.15 ± 0.01	3.48 ± 0.01
256	7.12 ± 0.16	6.09 ± 0.15	6.86 ± 0.18
512	$\mathbf{14.34 \pm 0.21}$	$\mathbf{12.54 \pm 0.23}$	$\mathbf{13.97 \pm 0.23}$
Denoising	2.60 ± 0.01	2.68 ± 0.01	2.60 ± 0.01

Table 3: Reconstruction GPU memory (GB) by spp

spp	Bedroom	Cornell Box	Kitchen
32	1.26	1.11	1.22
64	2.50	2.21	2.42
128	4.99	4.41	4.82
256	9.95	8.80	9.63
512	**19.89**	**17.57**	**19.25**

the largest model (512 neurons $\times$ 8 layers) yields only marginal improvements over our default configuration, while requiring significantly more training time to reach comparable quality. Conversely, smaller capacity models quickly plateau at distinctly lower performance levels.

Ablation Study

The encoding ablation table in the main paper presents an ablation study evaluating the individual contributions of our method's core components: the coordinate encoding, auxiliary pixel features (Aux.), and the denoiser. All configurations are trained for 100k iterations and evaluated across 2,000 lighting configurations per scene.

While using auxiliary features alone yields surprisingly strong metrics, this configuration inadvertently acts as an implicit denoiser; its simple identity position encoding lacks the capacity to capture high-frequency Monte Carlo noise, thereby artificially smoothing the output. We find it is much more principled and reliable to use a dedicated Monte Carlo denoiser (OIDN) combined with a multiresolution hashgrid encoding, which possesses the necessary capacity to learn high-frequency details.

As demonstrated in the table, directly training the high-capacity hashgrid on noisy Monte Carlo images causes the network to overfit to the noise. However, when the training targets are explicitly denoised first, our full model (Aux. + Enc. + Den.) consistently achieves the highest overall quality, outperforming the auxiliary features-only baseline across nearly all metrics.

Figure 4 shows visual comparison of the different components for an uncurated lighting configuration.

Production scene rendering cost

Table 4 reports the rendering time and memory consumption of a traditional path tracer on the three production scenes. These figures represent the cost an artist must pay each time the lighting is changed, before any preview becomes available at the given sample count. By contrast, our trained model loads in under a second, requires only a fraction of the memory, and produces relighting previews at interactive rates.

Table 4: Rendering time and memory usage for some production scenes. Data rendered at 512 spp (960×402), using 32 threads of an Intel Xeon Gold 6342 @ 2.8 GHz.

Scene	*Cave*	*Office*	*Tunnel*
Rendering time	28 m 12 s	30 m 46 s	35 m 37 s
RAM usage	28 GB	19 GB	14 GB

Additional Architecture Details

We encode the input pixel coordinate using a two-dimensional multi-resolution hashgrid [MESK22]. The encoding uses $L = 16$ levels with resolutions growing geometrically by a factor of $b = 1.3$ from $N_{\min} = 16$. At each level, the coordinate is bilinearly interpolated from a hash table of size $T = 2^{17}$ with $F = 2$ trainable features per entry. Features from all levels are concatenated into a 32-dimensional encoding that is passed to the subsequent MLP. We use the `tiny-cuda-nn` [Mül21] implementation compiled in single-precision floating point to avoid training instabilities.

References

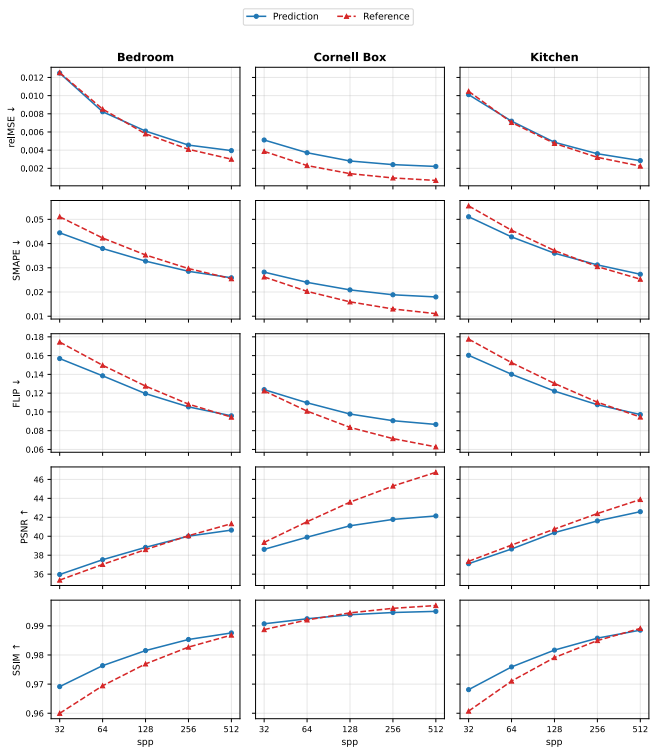
[Áfr26] ÁFRA A. T.: Intel® Open Image Denoise, 2026. https://www.openimagedenoise.org. 1

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph. 41*, 4 (2022), 102:1–102:15. doi:10.1145/3528223.3530127. 2

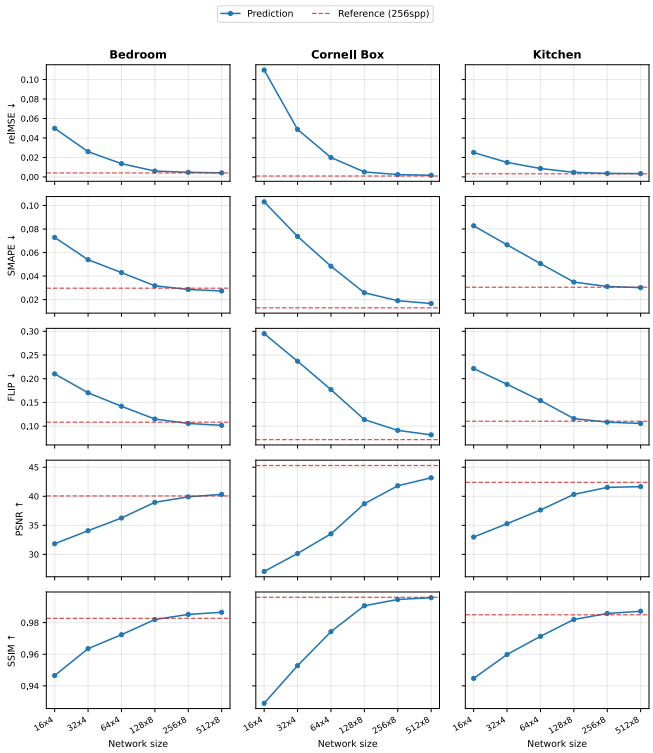
[Mül21] MÜLLER T.: tiny-cuda-nn, 4 2021. URL: https://github.com/NVlabs/tiny-cuda-nn. 2

Table 5: Quality metrics across different spp — prediction (Pred.) vs. Monte Carlo reference (Ref.), median over 2000 light configurations

Method	relMSE ↓		SMAPE ↓		FLIP ↓		PSNR ↑		SSIM ↑	
	Pred.	Ref.	Pred.	Ref.	Pred.	Ref.	Pred.	Ref.	Pred.	Ref.
bedroom / 32 spp	0.0125	0.0125	0.0444	0.0511	0.1569	0.1744	35.962	35.373	0.9691	0.9600
bedroom / 64 spp	0.0082	0.0085	0.0380	0.0423	0.1385	0.1497	37.524	37.026	0.9763	0.9694
bedroom / 128 spp	0.0061	0.0058	0.0327	0.0353	0.1195	0.1275	38.828	38.594	0.9815	0.9769
bedroom / 256 spp	0.0046	0.0041	0.0285	0.0297	0.1054	0.1083	40.010	40.057	0.9853	0.9827
bedroom / 512 spp	**0.0040**	**0.0030**	**0.0259**	**0.0255**	**0.0958**	**0.0945**	**40.648**	**41.326**	**0.9876**	**0.9868**
cbox / 32 spp	0.0051	0.0039	0.0282	0.0262	0.1237	0.1224	38.617	39.350	0.9907	0.9887
cbox / 64 spp	0.0037	0.0023	0.0240	0.0203	0.1097	0.1008	39.902	41.533	0.9924	0.9920
cbox / 128 spp	0.0028	0.0014	0.0209	0.0159	0.0977	0.0835	41.105	43.601	0.9938	0.9945
cbox / 256 spp	0.0024	0.0009	0.0189	0.0130	0.0906	0.0715	41.777	45.301	0.9946	0.9960
cbox / 512 spp	**0.0022**	**0.0007**	**0.0180**	**0.0111**	**0.0867**	**0.0627**	**42.143**	**46.748**	**0.9950**	**0.9970**
kitchen / 32 spp	0.0101	0.0105	0.0511	0.0556	0.1603	0.1776	37.104	37.355	0.9681	0.9607
kitchen / 64 spp	0.0072	0.0070	0.0428	0.0455	0.1401	0.1525	38.655	39.066	0.9759	0.9710
kitchen / 128 spp	0.0049	0.0047	0.0360	0.0371	0.1222	0.1303	40.383	40.759	0.9817	0.9791
kitchen / 256 spp	0.0036	0.0032	0.0312	0.0305	0.1077	0.1103	41.621	42.398	0.9858	0.9850
kitchen / 512 spp	**0.0028**	**0.0022**	**0.0273**	**0.0253**	**0.0972**	**0.0946**	**42.593**	**43.879**	**0.9885**	**0.9891**



(a) Samples per pixel



(b) Network size

Figure 1: **Quality metrics vs. samples per pixel (left) and vs. network size (right)** Prediction quality (blue) and Monte Carlo reconstruction reference (red) across three scenes. Left: Higher spp improves reconstruction quality and prediction quality. Interestingly, at low spp regimes, the model prediction can perform better than the data it was trained on. Right: Bigger models improve prediction quality. We use 256 neurons with 8 layers by default.

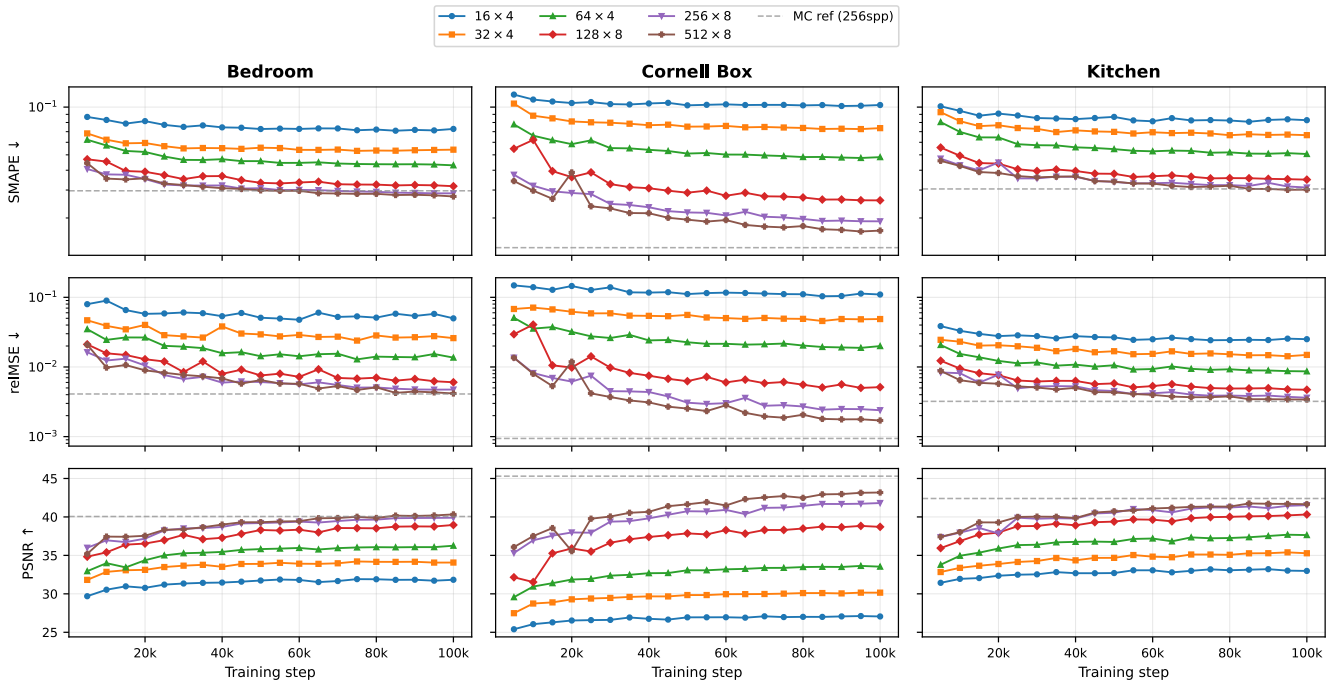


Figure 2: Convergence for different network sizes wrt. training step (100k)

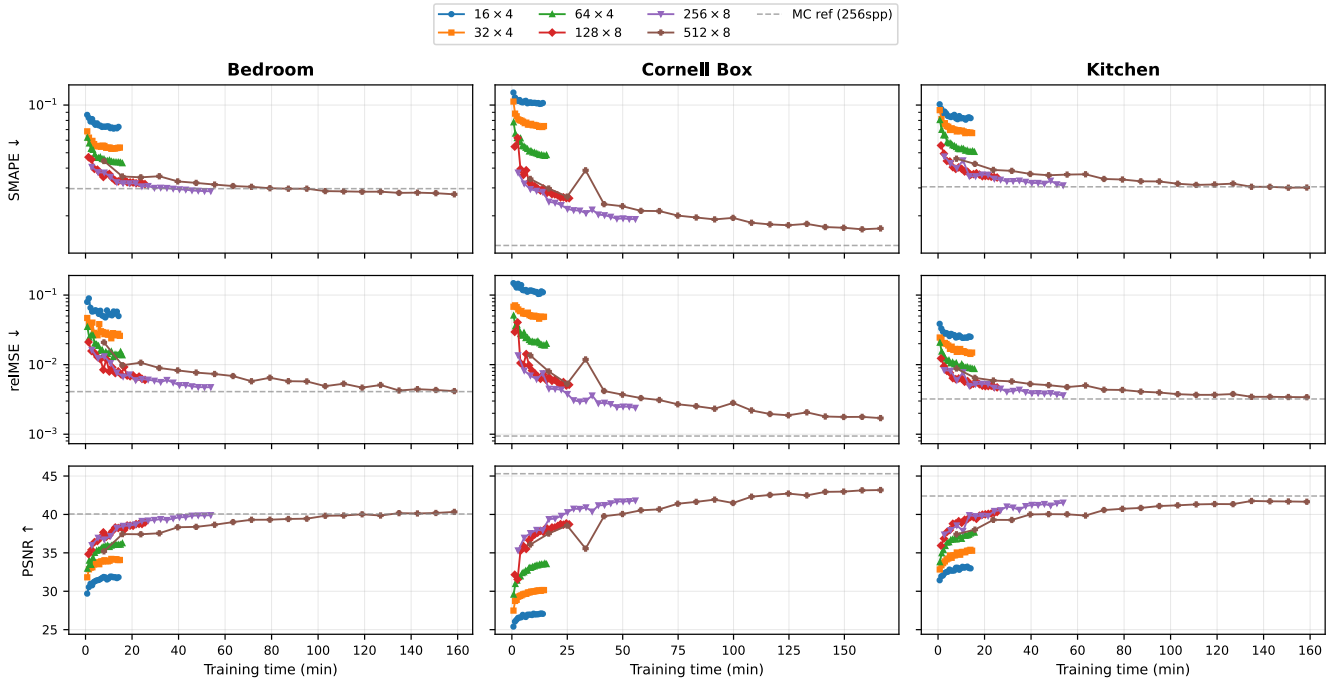


Figure 3: Convergence for different network sizes wrt. wall-clock time

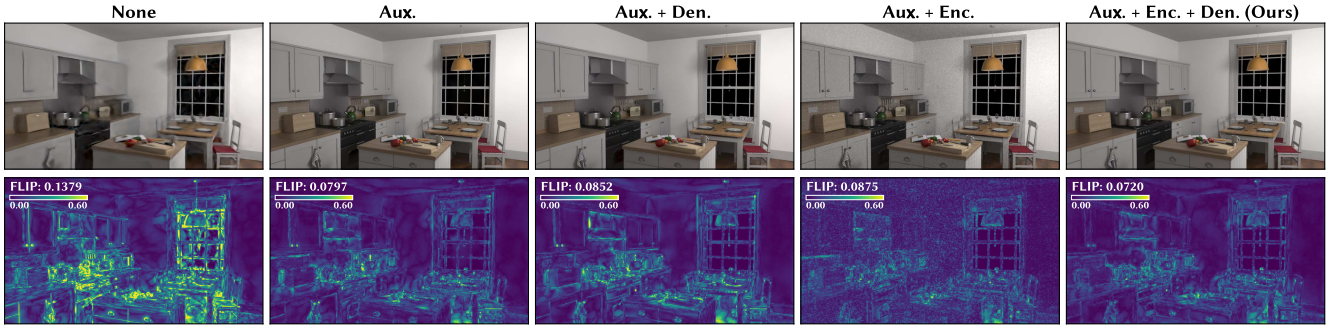


Figure 4: Qualitative comparison of component ablation variants on the Kitchen scene. Top row: tonemapped renderings under the same light position. Bottom row: FLIP error maps (HDR mode) against the ground truth, with shared scale. Our full pipeline (rightmost) produces the lowest perceptual error.

Table 6: Training profile – spp ablation

	Config	Fwd (ms / %)		Bwd (ms / %)		Opt (ms / %)		Data (ms / %)		Pool (ms / %)		Total (ms)	Est. 100k (min)	GPU (GB)
Bedroom	32 spp	**9.2**	31%	**18.7**	63%	**0.2**	1%	**0.1**	0%	**1.5**	5%	**29.9**	49.8	10.2
	64 spp	**9.2**	30%	**18.9**	62%	**0.2**	1%	**0.1**	0%	**1.9**	6%	**30.4**	50.8	11.6
	128 spp	**9.2**	30%	**18.9**	61%	**0.2**	1%	**0.1**	0%	**2.6**	8%	**31.2**	52.3	14.1
	256 spp	**9.2**	28%	**18.8**	58%	**0.2**	1%	**0.1**	0%	**4.0**	12%	**32.4**	54.7	19.0
	512 spp	**9.3**	26%	**18.8**	53%	**0.3**	1%	**0.1**	0%	**6.7**	19%	**35.3**	60.6	29.0
Cornell Box	32 spp	**9.6**	31%	**19.7**	63%	**0.2**	1%	**0.1**	0%	**1.5**	5%	**31.2**	52.1	10.6
	64 spp	**9.7**	30%	**19.8**	62%	**0.2**	1%	**0.1**	0%	**1.8**	6%	**31.7**	52.9	11.7
	128 spp	**9.7**	30%	**19.8**	61%	**0.2**	1%	**0.1**	0%	**2.4**	8%	**32.4**	54.2	13.9
	256 spp	**9.7**	29%	**19.8**	59%	**0.2**	1%	**0.1**	0%	**3.6**	11%	**33.5**	56.3	18.3
	512 spp	**9.8**	27%	**19.7**	55%	**0.3**	1%	**0.1**	0%	**6.1**	17%	**36.1**	61.3	27.0
Kitchen	32 spp	**9.2**	31%	**18.8**	63%	**0.2**	1%	**0.1**	0%	**1.5**	5%	**30.0**	50.0	10.3
	64 spp	**9.3**	30%	**18.9**	62%	**0.2**	1%	**0.1**	0%	**1.9**	6%	**30.4**	50.8	11.5
	128 spp	**9.3**	30%	**19.0**	61%	**0.2**	1%	**0.1**	0%	**2.5**	8%	**31.2**	52.2	13.9
	256 spp	**9.3**	29%	**18.9**	58%	**0.2**	1%	**0.1**	0%	**3.9**	12%	**32.4**	54.6	18.7
	512 spp	**9.4**	27%	**18.9**	54%	**0.3**	1%	**0.1**	0%	**6.5**	19%	**35.3**	60.4	28.3

Table 7: Training profile – network size ablation

	Config	Fwd (ms / %)		Bwd (ms / %)		Opt (ms / %)		Data (ms / %)		Pool (ms / %)		Total (ms)	Est. 100k (min)	GPU (GB)
Bedroom	16×4	**0.9**	11%	**3.4**	40%	**0.2**	2%	**0.1**	1%	**3.8**	45%	**8.5**	14.9	12.3
	32×4	**1.0**	12%	**3.6**	41%	**0.2**	2%	**0.1**	1%	**3.8**	43%	**8.9**	15.3	12.5
	64×4	**1.3**	13%	**4.0**	42%	**0.2**	2%	**0.1**	1%	**3.8**	40%	**9.5**	16.4	12.9
	128×8	**3.2**	21%	**7.9**	51%	**0.2**	1%	**0.1**	1%	**3.9**	25%	**15.3**	26.1	15.5
	256×8	**9.2**	28%	**18.8**	58%	**0.2**	1%	**0.1**	0%	**3.9**	12%	**32.3**	54.4	19.0
	512×8	**31.1**	33%	**59.4**	62%	**0.2**	0%	**0.1**	0%	**4.2**	4%	**95.1**	159.2	26.4
Cornell Box	16×4	**0.9**	11%	**3.6**	43%	**0.2**	2%	**0.1**	1%	**3.5**	42%	**8.4**	14.5	11.2
	32×4	**1.1**	12%	**3.7**	43%	**0.2**	2%	**0.1**	1%	**3.5**	40%	**8.7**	14.9	11.4
	64×4	**1.3**	14%	**4.2**	45%	**0.2**	2%	**0.1**	1%	**3.5**	37%	**9.4**	16.1	11.9
	128×8	**3.3**	22%	**8.2**	53%	**0.2**	1%	**0.1**	1%	**3.5**	23%	**15.5**	26.3	14.6
	256×8	**9.7**	29%	**19.8**	59%	**0.2**	1%	**0.1**	0%	**3.6**	11%	**33.4**	56.2	18.3
	512×8	**32.6**	33%	**62.8**	63%	**0.2**	0%	**0.1**	0%	**4.0**	4%	**99.8**	166.8	25.8
Kitchen	16×4	**0.9**	11%	**3.4**	40%	**0.2**	3%	**0.1**	1%	**3.7**	44%	**8.4**	14.6	12.0
	32×4	**1.0**	12%	**3.6**	42%	**0.2**	2%	**0.1**	1%	**3.7**	42%	**8.8**	15.2	12.2
	64×4	**1.3**	14%	**4.0**	43%	**0.2**	2%	**0.1**	1%	**3.7**	40%	**9.4**	16.2	12.6
	128×8	**3.2**	21%	**7.9**	52%	**0.2**	1%	**0.1**	1%	**3.8**	25%	**15.2**	26.0	15.2
	256×8	**9.2**	29%	**18.8**	58%	**0.2**	1%	**0.1**	0%	**3.9**	12%	**32.3**	54.4	18.7
	512×8	**31.1**	33%	**59.5**	63%	**0.2**	0%	**0.1**	0%	**4.1**	4%	**95.1**	159.1	26.0

Table 8: Quality metrics across different spp (inference mean)

Method	MSE ↓	relMSE ↓	LogMSE $\times 10^3$ ↓	SMAPE ↓	PSNR ↑	SSIM ↑	TonemappedMSE $\times 10^3$ ↓	FLIP ↓
bedroom / 32 spp	0.2998	0.4424	1.0375	0.0524	35.7308	0.9631	0.3427	0.1739
bedroom / 64 spp	0.2655	0.1786	0.7586	0.0434	37.3012	0.9724	0.2341	0.1500
bedroom / 128 spp	0.2471	0.1511	0.6379	0.0379	38.5196	0.9783	0.1821	0.1313
bedroom / 256 spp	**0.2383**	**0.1090**	0.5533	0.0333	39.7299	0.9827	0.1399	0.1166
bedroom / 512 spp	0.2440	0.2392	**0.5286**	**0.0312**	**40.3096**	**0.9852**	**0.1268**	**0.1091**
cbox / 32 spp	1.6337	**0.0875**	1.4725	0.0318	37.9438	0.9885	0.2248	0.1312
cbox / 64 spp	**1.5892**	0.2400	1.3772	0.0282	39.1339	0.9905	0.1887	0.1185
cbox / 128 spp	1.6263	0.2268	1.2759	0.0249	40.1979	0.9921	0.1570	0.1069
cbox / 256 spp	1.5912	0.2716	**1.2129**	0.0230	40.8486	0.9930	0.1402	0.1002
cbox / 512 spp	1.8547	0.2666	1.2197	**0.0221**	**41.1722**	**0.9934**	**0.1334**	**0.0971**
kitchen / 32 spp	0.1650	0.0222	0.9212	0.0599	36.9940	0.9622	0.2921	0.1787
kitchen / 64 spp	0.1226	0.0230	0.6984	0.0504	38.5283	0.9714	0.2135	0.1554
kitchen / 128 spp	0.1251	0.3941	0.6110	0.0438	39.8913	0.9776	0.1730	0.1390
kitchen / 256 spp	0.1109	0.0273	0.5363	0.0381	41.1432	0.9824	0.1444	0.1238
kitchen / 512 spp	**0.1109**	**0.0178**	**0.4677**	**0.0357**	**41.9245**	**0.9849**	**0.1240**	**0.1167**

Table 9: Quality metrics across different spp (inference median)

Method	MSE ↓	relMSE ↓	LogMSE $\times 10^3$ ↓	SMAPE ↓	PSNR ↑	SSIM ↑	TonemappedMSE $\times 10^3$ ↓	FLIP ↓
bedroom / 32 spp	0.0084	0.0125	0.6394	0.0444	35.9623	0.9691	0.2534	0.1569
bedroom / 64 spp	0.0066	0.0082	0.4663	0.0380	37.5236	0.9763	0.1769	0.1385
bedroom / 128 spp	0.0057	0.0061	0.3641	0.0327	38.8278	0.9815	0.1310	0.1195
bedroom / 256 spp	0.0053	0.0046	0.2794	0.0285	40.0104	0.9853	0.0998	0.1054
bedroom / 512 spp	**0.0048**	**0.0040**	**0.2624**	**0.0259**	**40.6482**	**0.9876**	**0.0861**	**0.0958**
cbox / 32 spp	0.0390	0.0051	0.6356	0.0282	38.6174	0.9907	0.1375	0.1237
cbox / 64 spp	0.0367	0.0037	0.5157	0.0240	39.9018	0.9924	0.1023	0.1097
cbox / 128 spp	0.0340	0.0028	0.4325	0.0209	41.1046	0.9938	0.0775	0.0977
cbox / 256 spp	0.0332	0.0024	0.3724	0.0189	41.7772	0.9946	0.0664	0.0906
cbox / 512 spp	**0.0332**	**0.0022**	**0.3675**	**0.0180**	**42.1427**	**0.9950**	**0.0611**	**0.0867**
kitchen / 32 spp	0.0009	0.0101	0.3631	0.0511	37.1044	0.9681	0.1948	0.1603
kitchen / 64 spp	0.0006	0.0072	0.2489	0.0428	38.6549	0.9759	0.1363	0.1401
kitchen / 128 spp	0.0004	0.0049	0.1665	0.0360	40.3829	0.9817	0.0916	0.1222
kitchen / 256 spp	0.0003	0.0036	0.1226	0.0312	41.6205	0.9858	0.0689	0.1077
kitchen / 512 spp	**0.0003**	**0.0028**	**0.1019**	**0.0273**	**42.5929**	**0.9885**	**0.0550**	**0.0972**

Table 10: Quality metrics across different network sizes (median)

Method	MSE ↓	relMSE ↓	LogMSE $\times 10^3$ ↓	SMAPE ↓	PSNR ↑	SSIM ↑	TonemappedMSE $\times 10^3$ ↓	FLIP ↓
bedroom / 16×4	0.0835	0.0499	2.7739	0.0728	31.8276	0.9465	0.6565	0.2102
bedroom / 32×4	0.0609	0.0260	1.6851	0.0539	34.0698	0.9635	0.3918	0.1704
bedroom / 64×4	0.0290	0.0136	0.9577	0.0430	36.2580	0.9723	0.2367	0.1419
bedroom / 128×8	0.0087	0.0060	0.4049	0.0317	38.9556	0.9820	0.1272	0.1150
bedroom / 256×8	0.0051	0.0047	0.2931	0.0286	39.8978	0.9852	0.1024	0.1057
bedroom / 512×8	**0.0040**	**0.0042**	**0.2501**	**0.0273**	**40.3297**	**0.9865**	**0.0927**	**0.1017**
cbox / 16×4	0.9636	0.1098	12.1776	0.1031	27.0584	0.9290	1.9686	0.2951
cbox / 32×4	0.7320	0.0488	6.6656	0.0737	30.1532	0.9528	0.9653	0.2370
cbox / 64×4	0.2971	0.0199	3.0939	0.0484	33.5351	0.9743	0.4431	0.1770
cbox / 128×8	0.0725	0.0051	0.8273	0.0258	38.7090	0.9907	0.1346	0.1138
cbox / 256×8	0.0316	0.0024	0.3833	0.0191	41.8071	0.9946	0.0660	0.0911
cbox / 512×8	**0.0191**	**0.0017**	**0.2510**	**0.0166**	**43.1770**	**0.9958**	**0.0481**	**0.0816**
kitchen / 16×4	0.0024	0.0251	0.9458	0.0827	32.9795	0.9447	0.5036	0.2215
kitchen / 32×4	0.0016	0.0149	0.5947	0.0665	35.2762	0.9598	0.2967	0.1882
kitchen / 64×4	0.0009	0.0086	0.3360	0.0506	37.6407	0.9713	0.1722	0.1540
kitchen / 128×8	0.0005	0.0047	0.1725	0.0349	40.3181	0.9820	0.0929	0.1160
kitchen / 256×8	0.0003	0.0036	0.1255	0.0311	41.5303	0.9858	0.0703	0.1084
kitchen / 512×8	**0.0003**	**0.0034**	**0.1237**	**0.0302**	**41.6527**	**0.9872**	**0.0683**	**0.1057**